

APOLOGIES are in order to all beacause I didn't give the games REVERSE and SIMON a good going-over before printing them, and there are a number of errors. Since then, Brett has also managed to simplify SIMON, and the corrections to both appear later on. By the way, unless you fellows and gals are really desperate, I think that questions, etc., by mail would be more completely answered by the contributors instead of by telephone.

SYMBOLS that I have been using may have caused some confusion. Firstly, I slid into using an asterisk \* for the multiplication sign in the hand-written programs last issue, forgetting that there is an asterisk symbol on the keypad. From now on I will use a small x for multiplication. Next, the symbol # is sometimes used for the 'not equals' or  $\neq$  (because it takes only one key stroke of the typewriter). In the Bally, this does mean 'not equals' when preceded by IF, and it is also used to describe a format convention for the tabulation function when preceded by PRINT. Symbol 0 is used for the numeral zero to avoid confusion with the letter 'O'. Symbol 7 is sometimes used for the numeral seven.

UTILIZATION of the Bally by one of our subscribers is unique. They operate a TV booster system, capturing long distance TV signals on mountain tops and rebroadcast them into valleys that normally lie in a 'shadow'. The Bally is used to insert 'commercials'. and programs are developed using the graphics capability to generate logos, and the &(9) and &(10) are used for screen wipes, color changes, etc.

SEMINARS are being planned at a couple of locations, being developed by local dealers. One is being considered in Indiana where all levels of users would be accommodated, while the other is a bit farther along. This group will be having a get-together at 2pm on May 12 at the Computer Center, 28251 Ford Rd. Garden City, MI, (422-2570).

TRANSLATIONS of the various BASIC dialects are contained in the new book, The BASIC Handbook by David Lien, published by CompuSoft Publishing Co., P.O. Box 19669, San Diego, CA 92119, for \$14.95 + 1.35 post + CA tax. I understand that it has 250 statements in BASIC with their meanings, plus conversion ideas to other dialects. There is a review in the April issue of Creative Computing, p.143; and an ad in May Kilobaud, p.81.

PROJECTS Who is working on what - and what is your status? We'll get some of you together so's you only invent the wheel once. Have any of you been able to use another computer's printing facility to list out the Bally programs?

MENU as mentioned last time was brought up with a little program, but I've had notes that plain CALL 3172 or CALL 3177 will do it.

TUTORIALS (Hows and whys) are needed in the ARCADIAN in order that we can learn more about the machine and its operation. The talents of our subscribers run the gamut from the tyro to the professional, and for many of us it is necessary to do things in a cook-book manner, not understanding what we are doing or why. Explanations such as those that follow are going to be of great value to us as we plod ahead.

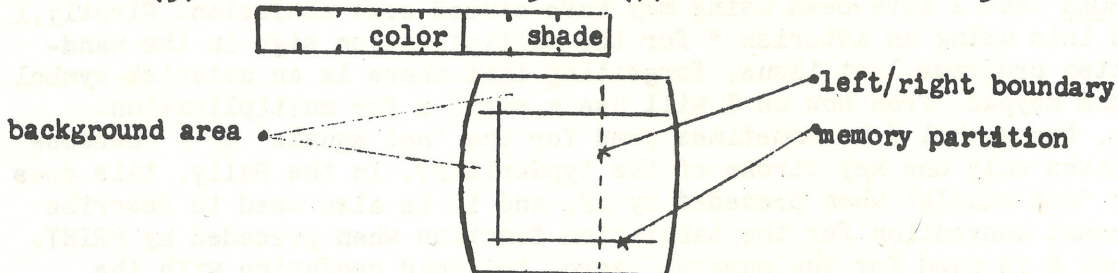


# arcadian

## TUTORIAL (1) SCREEN OPERATIONS, by John Perkins.

The screen is divided into a left and a right side with a movable boundary. The following outputs prevail:

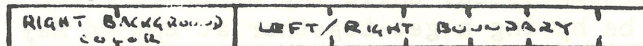
&(0)=right side register 0	&(4)=left side register 0
&(1)= " " " 1	&(5)= " " " 1
&(2)= " " " 2	&(6)= " " " 2
&(3)= " " " 3	&(7)= " " " 3



Bally BASIC continually sets &(4) and &(5) to the color/shade defined by BC, and &(6) and &(7) to the color/shade defined by FC. These are fixed while BASIC is in control. But by moving the boundary so that the right side is visible, we can then control 4 different color/shades by using the &(0) thru &(3). Example:

&(0)=30;&(1)=85;&(2)=153;&(3)=125;&(9)=0

Three colors are displayed- listing, background, and 'garbage' at the top. More on this later. The fourth color should be visible as we scroll the text into the upper border area. With &(9) at some other value, such as 10, the screen is divided and the FC and BC commands allow two more colors on the screen. Actually, &(9) has two functions:



The least significant 6 bits set the boundary position (4 pixels or one memory byte per unit). The most significant 2 bits choose the color register associated with the left side background and the right side background. Try &(9)=135.

The 'garbage' mentioned above is actually the stored program, in the screen memory, using the even bit positions.

Each pixel equates to two bits of memory - 4

pixels to an 8 bit byte. The two bits of each pixel can have 4 representations;

00 = &(4) left	or	&(0) right
01 = &(5) left	or	&(1) right
10 = &(6) left	or	&(2) right
11 = &(7) left	or	&(3) right

When Bally BASIC sets the screen boundary ( &(9) ) all the way to the right, then only the left registers &(4) to &(7) are used. Since it also sets &(4) and &(5) to BC, and &(6) and &(7) to FC, only the odd bits of memory show on the screen.

a 00 is the same color as an 01, and a 10 is the same color as a 11 By storing the program in the even bits it can occupy screen memory ( as every other bit) and yet be invisible. However, by moving the boundary to the left, the right-side registers are used, and since the program above set these to different colors, the stored program becomes "visible" as the garbage at the top of the screen.

&(10) controls how many raster lines are displayed from memory as opposed to being part of the background. &(10)=204 displays all of memory (RAM) allowing visual inspection of the running program.



## TUTORIAL (2) EXPLANATIONS by Jean Taillefer.

- IF statements will execute as a TRUE condition if the value of the expression results in a value greater than 0. (The expression could be a calculation) If the value is 0 or negative, the IF statement regards the condition as FALSE.  
 Example: IF TR(1) GOTO 100 . will branch to 100 if TR(1) is = 1  
           IF A GOTO 110 . will branch to 110 if A is greater than 0
- AND may be expressed in many ways. The most common form being  
       IF A=3 IF B=0 GOTO 120 . will branch to 120 if and only if A=3 and if B=0  
       IF(A=3)=(B=0) GOTO 120 . does the same thing
- OR conditions, where you want to jump if either of some conditions are true, can be done by  
       IF A=3 GOTO 130  
       IF A=6 GOTO 130  
       IF A=7 GOTO 130 .meaning that if A is either 3,6,or 7, the program will jump to 130  
       IF(A=3)+(A=6)+(A=7) GOTO 130 . does the same thing
- Self-starting programs can be made by inserting the line  
       1: RETURN at the beginning, and use this at the end to store on cassette- NT=1; :PRINT:LIST:PRINT"CLEAR;RUN"

## TUTORIAL (3) DATA STORAGE by Bob Weber.

This subroutine would be called up in order to save the program, the registers, and the strings by using a GOTO 9000.

```

9000 :PRINT:LIST
9010 FOR Z = 1 TO 26 .less,if you don't have that much
9020 TV = Z + 64 .register storage
9030 PRINT #1, $(20076+(Zx2))
9040 NEXT Z
9050 FOR Y = 0 TO N .where N is the number of strings
9060 PRINT #1, "a(",Y,")=",a(Y)
9070 NEXT Y
9080 PRINT "RUN"
```

CHECKERS GAME listing by John Collins, 713 Bradford Drive, Ft. Walton Beach, FL 32548 is included. There is an amazing amount of activity in this game, that is comparable to the \$75. 'Checker Challenger'. Before the machine makes a move, it goes thru some steps, and numbers appear to tell you where it is. The code for the steps is:

1. the computer has found that it can jump one of your men
2. checking to see if you can jump it
3. is a corner open?
4. is there an open move?
5. & 6. have the computer's men moving either to get kinged or towards and player's man left
7. any move an unkinged computer's piece can make
8. any move

To indicate a double jump, enter the two numbers (of the square you go thru and the landing square) as if it were a single jump only.

# PROGRAM NAME CHECKERS

```

Line #      Statements
4          I RETURN; CLEAR
6          PRINT "TO(C)CHECKERS+JOHN
8          COLLINS"; GOSUB 3000
50         FOR U=A TO B; IF @ (U) < 4
55         NEXT U; FOR V=1 TO 10; PRINT
V; FOR U=A TO B; IF @ (U) < 4
60         FOR X=1 TO 1 STEP 2; FOR Q=1
TO 1 STEP 2; IF @ (U) = 4 Q=-1
70         B=Q * X; S=U+B; IF @ (S) = 0
GOTO 900
75         IF V=2 IF @ (S) # 3 GOTO 900
80         C=B+B; F=Q+2; Z=Y+100;
J=0; GOSUB 3000; IF J=1 GOTO 900
90         GOTO 900
100        IF @ (U+C) # 3 RETURN
200        IF @ (S) - 2; RETURN
220        IF V=1 S=U+C; J=1; RETURN
230        F @ (U-B) # 3 RETURN
240        IF @ (U-F) > 3 S=U-B; U=U-F;
J=1; RETURN
250        IF @ (U-C) > 3 S=U-B; U=U-C;
J=1; RETURN
260        IF @ (U-B-F) = 3 S=U-B-F;
J=1; RETURN
290        RETURN
300        IF @ (S-F) # 1 RETURN
310        IF @ (S) = 3 J=1; RETURN
320        IF @ (S-Z) (X-10) = 3
S=5-2 * (X-10); J=1
330        RETURN
400        IF @ (U+C) = 0 IF @ (U) = 4 J=1
410        RETURN
500        IF @ (U+C) < 3 RETURN
510        IF @ (U+F) < 3 IF @ (U+C-F)
= 3 RETURN

```

# PROGRAM NAME

```

Line #      Statements
520        IF @ (U+C-F) = 1 IF @ (U+F)
= 3 RETURN
530        GOTO 700
600        L=1; IF @ (U+C) - 3 < 0 RETURN
610        IF @ (U+F) - 3 < 0 IF @ (U+C-F)
= 3 RETURN
620        IF @ (U+F) = 3 IF @ (U+C-F) = 1
RETURN
630        L=L+1; D=@ (U+L-B); IF
D=0 RETURN
640        IF D-3 > 0 RETURN
650        IF L=2 IF D-3 < 0 J=1; RETURN
660        GOTO 630
700        IF @ (U) = 5 RETURN
800        J=1; RETURN
900        NEXT Q; NEXT X
910        NEXT U; NEXT V
920        BC=8; GOSUB 2000; PRINT
YOU WIN"; A=KP; GOTO 8
960        R=U; Q=-1; NEXT Q;
X=1; NEXT X
970        V=0; U=BQ; NEXT U; NEXT V;
T=-1; GOTO 1200
1000       PRINT "R"; R; "S";
INPUT "FROM"; IF S < A A=S
1010        IF @ (R) > 2 GOTO 1000
1020        IF @ (R) = 0 GOTO 1000
1030        INPUT "TO"; S; IF @ (S) # 3
GOTO 1000
1040        IF @ (R) # 1 IF R > S GOTO 1000
1200        IF @ (S-R) * (S-R) < 122
GOTO 1000
1210        IF T > 0 IF @ (S+R) > 2 < 4
GOTO 1000
1220        J=T; @ (S) = @ (R);
@ (S+R) + 2 = 3; @ (R) = 3
1400        FOR X=1 TO 1 STEP 2
1410        IF @ (S+J * X) = 3 + T GOTO 1450
1420        IF @ (S+J * X) = 3 + T + T
GOTO 1450

```

# PROGRAM NAME

```

Line #      Statements
1430        GOTO 1500
1450        IF @ (S+2 * J * X) # 3 GOTO 1500
1460        R=S; S=S+2 * J * X
1470        X=1; NEXT X; GOTO 1220
1500        NEXT X; IF @ (S) - 3 = 2
IF J=1 J=-1; GOTO 1400
1510        GOTO 1610
1600        @ (S) = @ (R); @ (R) = 3
1610        IF T > 0 IF S > 0 @ (S) = 1
1620        IF T < 0 IF S < 0 @ (S) = 5
1630        IF T > 0 GOTO 500
1640        T=1; GOSUB 2000; GOTO 1000
2000        CLEAR; BOX 250, 96, 86, 3
FOR I=1 TO 8; IF @ (I) = 0
GOTO 2100
2020        M=-25 * (I - (I+1)) * 10
2030        N=-45 * (I+1) * 10
CX=M-12; CY=N; PRINT #2, I
1000        IF @ (I) # 3 BOX M, N, 7, 2, 1
2060        IF @ (I) > 3 BOX M, N, 2, 2, 3
1000        IF @ (I) - 3 = 2 BOX
M, N, 7, 4, 3
2100        NEXT I; RETURN
3000        FOR I=1 TO 100; @ (I) = 0;
NEXT I; FOR I=12 TO 18
STEP 2
@ (I) = 2 * @ (I+1) = 2; @ (I+2) =
2; @ (I+3) = 3; @ (I+4) = 3
@ (I+5) = 4; @ (I+6) = 4;
@ (I+7) = 4; NEXT I; A=67
BC=7; FC=146; RETURN

```

DO NOT ENTER A SPACE BETWEEN LINE # AND STATEMENT THIS IS DONE BY THE UNIT



**MEMORY DUMPS and LOADERS** I have received about 6 programs that 'dump' the data located in the ROM in various languages, and one is included herein that prints its answers in binary. As the others get 'scrubbed', they will be included for your information. What to do with the knowledge you will then have is up for debate.

**MEMORY DUMP** listing was written by Max Manowsky to yield a binary output for a selected memory location. Brett Bilbray has modified it to give a full 16 bit answer, and added the comments to go with it.

Line #	Statements	Comments
1	MEMORY CONTENTS-BINARY	
2	BY MAX MANOWSKI	
3	MODIFIED BY B. BILBRAY	
10	INPUT D; CLEAR; PRINT#7, D;	
	A=%(D); PRINT#7, A;	
	IF A(0) GOTO 30	
20	GOSUB 1000; GOTO 2000	
30	A=-A; GOSUB 1000; FOR B=1 TO	
	16; IF @ (B)=48 @ (B)=49; GOTO 50	
40	@ (B)=48	
50	NEXT B	
60	B=1	
70	@ (B)=@ (B)+1	
80	IF @ (B)=50 @ (B)=48;	
	B=B+1; GOTO 70	
90	GOTO 2000	
1000	FOR B=1 TO 16; @ (B)=A-A+2*2	
	+48; A=A+2; NEXT B; RETURN	
2000	FOR B=16 TO 1 STEP -1; IF (B=12)	
	+(B=8)+(B=4) TV=32	
2010	TV=@ (B); NEXT B; PRINT;	
	GOTO 10	

USE OF SHAD0 AREA IS FOR TWO OR MORE LINES OF MULTIPLE STATEMENTS

line 10 asks for the input for the desired location, the machine prints that location and then the PEEKed decimal number  
 line 20 calls for a conversion from decimal to binary, and displays binary  
 line 30-90 calls the decimal to binary conversion, performs a 'ones compliment' on the number and calls the display routine  
 line 1000 stores the decimal number as a binary in @ (X)  
 lines 2000, 2010 provides the display routine for the binary number

What you will get looks like this arbitrary example:

The location I requested is 2049

2049

5727

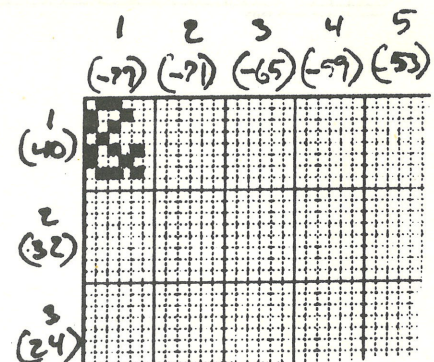
.decimal

0001 0110 0101 1111 .binary

**GRAPH** to the right is a portion of a worksheet being developed by Chuck Thomka to identify each pixel's location on the screen for details of figure construction. Note the ampersand &

Contact Chuck at 1228 West 222 St.,  
 Torrance, CA 90502

for information as to availability of this worksheet as well as the listing sheets as I use (sample-partial-above)





MEMORY DUMP program by Gary Moser prints its answers in Hexadecimal language. See what you get for the answer to location 0006. If it is 61, then your machine is like mine, and if it is 66, it is like Tom Wood's. If something else, then we have more variants on the street.

RANDOM ART is a quick little moving box program by Ernie Sams.

Line #      Statements

```

1  .RANDOM ART
2  .BY E. SAMS
10  X=0;Y=0
20  INPUT "WIDTH INCREMENT" W
30  INPUT "HEIGHT INCREMENT" H
40  X=X+W;Y=Y+H
50  CLEAR
60  IF X>159 W=-W;FC=RND
    (31)*0+4
70  IF X<2 W=-W
80  IF Y>79 H=-H
90  IF Y<2 H=-H
100  X=X+W;Y=Y+H
110  IF X<1 X=1
120  IF Y<1 Y=1
130  BOX 0,0,X,Y,3
140  GOTO 60

```

DO NOT ENTER A SPACE BETWEEN LINE # AND STATEMENT. THIS IS DONE BY THE UNIT

Line #      Statements

```

1  .MEMORY CONTENTS-HEX
2  .BY G. MOSER
5  GOTO 200
6  C=0
10  IF A<0 GOTO 120
20  FOR N=1 TO 4
30  B=A+16
40  IF RM<10 GOTO 60
50  RM=RM+7
60  @ (5-N)=RM+48
70  A=B
80  IF C=0 GOTO 90
81  A=A+2048
90  NEXT N
100  TV=@ (3)
101  TV=@ (4)
102  PRINT #1," ";
103  TV=@ (1)
104  TV=@ (2)
105  PRINT
110  RETURN
120  A=32767-ABS(A)+1
130  C=1
140  GOTO 20
200  PRINT "MEMORY LOCATION"
210  INPUT T,U
220  FOR V=T TO U STEP 2
230  A=% (V)
235  PRINT #0,V,
236  PRINT #1,"/",
240  GOSUB 6
250  NEXT V
260  GOTO 200

```

DO NOT ENTER A SPACE BETWEEN LINE # AND STATEMENT. THIS IS DONE BY THE UNIT



# arcadian

MACHINE LANGUAGE PROGRAMMING A further step along the way was taken by Glenn Pogue, who modified the "game over" routine of p. 25, making it print the word ARCADIAN in 2x normal letter size. I have not been able to totally duplicate this feat, I think it lies in the small differences in ROM locations that have previously been noted. The total program is:

<pre> 9 CLEAR 10 A=20180;B=A;C=120 20 X=-43;GOSUB C 30 X=53;GOSUB C 40 X=27672;GOSUB C 50 X=20190;GOSUB C 60 X=-13871;GOSUB C 70 X=21057;GOSUB C 80 X=16707;GOSUB C 90 X=18756;GOSUB C 100 X=20033;GOSUB C 110 CALL (B); STOP 120 %(A)=X; A=A+2; RETURN           </pre>	<p>.ref p.34"LINE INPUT BUFFER from 20180..."</p> <p>.lines 20 to 60 call subroutine 52 and define the required parameters for the 70 to 100 part to work, and get back to the BASIC</p> <p>.lines 70 thru 100 insert the letters per the scheme shown below</p> <p>.Displays the contents of memory slots A</p> <p>.POKES the values of X into memory slots A</p>
--	--

To convert the word ARCADIAN into machine language, each character is converted into its hexadecimal equivalent (use chart on p.16.) They are then paired off, each pair is swapped, and the new pair converted into decimal, as follows:

Desired characters	A	R	C	A	D	I	A	N
Hexadecimal conversion	41	52	43	41	44	49	41	4E
Pair off	41	52	43	41	44	49	41	4E
Swap within pair	52	41	43	41	44	49	41	4E
Convert to decimal(use routine on page 36)	21057	16707	18756	20033				

And these are the values of X in lines 70 to 100.

My operation did not give a clear display. There was more material on the screen, some of it seemed to be overprinting. I inserted line 105, X=12336; GOSUB C, to add some known characters (00), and I could then see the first part of line 20 → 20 X = -43 in giant letters.

The program is presented for the experimenters in the audience who would like to have something more unusual. The program has more potential because the root subroutine, 52, has many capabilities.

LETTERS from ARCADIAN subscribers to Bally, detailing what their desires would be in the capability of the Programming Keyboard might help the Bally management to move ahead on this project. The Director of Sales is Mr. J.Nieman, Bally Consumer Products Div., 10750 West Grand Ave. Franklin Park IL, 60131.

**SIMON CORRECTIONS:** Make the following changes in the program:

Revise line 10 CLEAR;&(0(=7;&(1)=7;&(2)=0;&(3)=0;&(9)=30;  
 NT=0;CX=47;CY=20;PRINT"SIMON";B=7;A=0;CX=47;  
 CY=-20;PRINT "SCORE:";NT=5

Delete lines 70,80,90,100,160,170

Add lines 70 FOR X=1 TO A

80 GOSUB @ (X)x1000

155 IF D=1 GOSUB 1000; GOTO 170

160 IF D=2 GOSUB 2000; GOTO 170

164 IF D=3 GOSUB 3000; GOTO 170

166 GOSUB 4000

Revise line 150 IF D#@ (X) NT=55;MU=33;MU=48;MU=48;NT=3;  
 FC=0;GOTO10

In lines 1000,2000,3000,4000 delete the -2xA after 1 TO 255



## ADS

Six programs available: Horserace, \$3.; TicTacToe, \$1; Craps 2, Startrek, Slot Machine, Connect Four, at \$2. each. All six for \$10. Include a C-30 tape for programming. Or listing for half price. All games except Startrek have graphics. S.Waldinger, 24740 Woodcroft Dr, Dearborn MI 48124

Conversions from Hex to Decimal, Decimal to Hex, and Binary to Hex and Decimal. All on one tape for \$5. Robert Strand 10665 E. FOIX Ave. Norwalk, CA 90650

The listing for Bob Weber's ad last month should have been: Bob Weber 6594 Swartout Rd. Algonac MI 48001 has the following available for \$2. each plus a tape long enough to accept 4 minutes per program. Or \$3. each on Bob's tape.

SUB SEARCH	ALIEN PATROL	CALENDAR
SLOT MACHINE	CONCENTRATION	TIC TAC TOE
FLIGHT SIMULATOR	HANGMAN	MATH QUIZ
OTHELLO	MASTERMIND	SPACE CHASE

A total of 21 games are available from Jean Taillefer, 115 Northwestern Ave. Ottawa, K1Y 0M1 Canada, at costs of \$1 for one minute, \$2. for three, and \$3. for a five minute program (you supply the tape). Or the listing is half price. Send for a list of those available.

ARCADE plus 'cades: 2002, 2003, 2004, 3001, 3002, 5002, plus DEMO Basic and DEMO cassette interface. (these will not do the tricks we talk about) total \$300. D. Choinisky, 1748 Wiese Ln, Racine WI 53406 414-886-9316

Two sets of programs available: Set I GAMES- Cheese Boxes, Random, Siren, Slot Machine, Color Match, Rock/Paper/Scissors, Memory Match, Building Blox Set II VIDEO ART- Wallpaper, Rnd Line, Rnd Box, Color Box, Scroll 1, 2, 3, Electric Dolly, Color War, Color Wheel, RubberBand, Laser Duel, Spiral, Reverse Box, Perspective Box. Prices are On His Cassette, \$8/Set or \$10/both

On Your Cassette, \$4/Set or \$6/both from

D. Stocker 333 Coronado Dr MtVernon, IN 47620

## REVERSE CORRECTIONS

Revise line 260 CY=-20;PRINT" YOU WON IN";TV=T;10+48;  
TV=T-T;10x10+48;PRINT "MOVES"  
270 GOT010  
280 CX=-50;CY=0

-46-

## ARCADIAN

Robert Fabris, proof reader  
3626 Morrie Dr.  
San Jose, CA 95127

FIRST CLASS